PATENT CLAIMS

1. The method of software emulation of the EEPROM memory in another non-volatile memory **characterized in that** after initiating the emulation, two sectors of the non-volatile memory are reserved, serving the function of the current sector and the auxiliary sector and two buffers are created in the operational memory, the first of which stores always the most current image of the emulated EEPROM memory, and the second stores the last patch, moreover the current sector of non-volatile memory is organized in such a way that a part of the sector contains the original image of the emulated memory, and remaining part is successively filled in with the patches, describing changes in the content of the original image of the emulated memory, in turn, at the time, when a new patch cannot be appended to the current sector, the functions of the sectors of non-volatile memory are changed, thanks to which the previously auxiliary sector of non-volatile memory is activated by saving the current image of the emulated memory from RAM memory to newly activated sector as a new original image of the emulated memory, however, after a correct writing, the content of the previously current sector of non-volatile memory is erased.

2. The method of emulation of the EEPROM memory according to claim 1, **characterized in that** the third buffer is created in the operational memory, which is used for storing the patch after compression.

3. The method of emulation of the EEPROM memory according to claim 2, **characterized in that** at initiating of the emulation the current sector is selected, next the content of the auxiliary sector is erased and next the original image of the emulated EEPROM memory is fetched from the current sector of the non-volatile memory to the operational RAM memory, next, the first patch is fetched from the current sector of non-volatile memory and its validity is checked, while if the patch was not invalidated it is checked if the patch data are compressed and in case when they are uncompressed, they are saved in the buffer of operational RAM memory, which stores the most current image of the emulated EEPROM memory, however, in opposite case, before saving to the memory buffer, the patch is decompressed, while in case, when the patch is invalidated it is skipped, while in the last point of the procedure of initiating emulation it is checked if there are still data for reading in the non-volatile memory and if there are such data, the next patches are processed according to the described algorithm.

4. The method of emulation of the EEPROM memory according to claim 3, **characterized in that** the current and the auxiliary sector is selected by setting the first sector as the current one, next it is checked if the saved data have a correct format and if the data were correctly saved, and if in case of such check it appears that there are incorrect data in the first sector, the second sector is set as the current sector, while the first sector is set as the auxiliary, however, in the opposite case, that is when there are correct data in the first sector, the second sector is set as the current one, and next it is checked if the saved data have a correct format and if the data were saved correctly, and if as a result of such check it appears that there are incorrect data in the second sector, the first sector is set as the current sector, while the second sector is set as the auxiliary one, while, in the

opposite case the sector, in which there is more free space, is set as the current sector.

5. The method of emulation of the EEPROM memory according to claim 2, **characterized in that** the writing process a new patch starts from preparing data for updating the content of the non-volatile memory, while the data are also saved in the buffer of the operational RAM memory, and next it is checked if the size of the patch is greater than the set value and if it is greater, the patch is compressed, then it is checked if the result of compression corresponds to the required assumptions of reducing the patch size, whereas in case of compliance the compressed patch is further processed, and in the opposite case the uncompressed patch is processed, next, it is checked if in the current sector of non-volatile memory there is sufficient space for saving the new patch and in case of sufficient space, the patch is saved, and in case of a lack of sufficient free space the current sector is changed into the second one and the new original image of the emulated EEPROM memory is saved in the second sector of non-volatile memory, while the content of the second sector in the Flash memory is erased.

6. The method of emulation of the EEPROM memory according to claim 5, **characterized in that** in the process of preparing the patch of the content of the non-volatile memory a preparation of the patch for saving in the memory is made, next the data are saved in the RAM memory, which stores the most current image of the EEPROM memory, at the same time the prepared patch is stored in the buffer of the RAM memory, and next it is checked whether the patch, which was last saved in the non-volatile memory is valid while, if the patch is invalidated the writing process the new patch is continued, while in case when the previously saved patch is valid, it is checked if the currently processed patch reverses the

changes introduced by the saving of the previous patch, whereas, if the processed patch did not reverse these changes, the writing process of the new patch is continued, and, in the opposite case, when the patch reverses the changes introduced by the saving of the previous patch, the value of the bit, which invalidates previously saved patch, is changed and the saving of the new patch to the non-volatile memory is canceled.

7. The method of emulation of the EEPROM memory according to claim 2, **characterized in that** the format of the patch consists of four fields, the first of which is the patch header, the second field appearing in case of patches containing many groups of data is a field of the size of data group, the third one appearing only in case of uncompressed patches is an offset field of data in relation to the initial address, while the last field in the patch is data field, while the patch contains many data groups, of which every one is saved under a different memory address, and the values of the offset of patch data groups contain an offset in relation to the final address of the previous data group, of which only the first offset defines the absolute address, while the next values are relative addresses in relation to the previous data group.

8. The method of emulation of the EEPROM memory according to claim 7, **characterized in that** the compressed patch does not contain the offset field, while the value of the offset is read only after decompression of the patch.

9. The method of emulation of the EEPROM memory according to claim 7, **characterized in that** the format of the header of the patch consist of the start bit, changed at the time when preparation of the patch for saving is started, the bit of correct writing of the size and format, changed after the size value and format of data are correctly recorded, the bit of correct writing, changed after the whole

patch is correctly recorded in the non-volatile memory, the invalidation bit, which is changed after the patch is invalidated and the field defining the format of the patch and the field, defining the total amount of data in the patch, not considering the size of the header.

10. The method of emulation of the EEPROM memory according to claim 9, **characterized in that** the format of the patch is defined as one of three types, that is as the format of a single update, or as the multi data groups patch or as the compressed patch.

11. The method of emulation of the EEPROM memory according to claim 5, **characterized in that** for the format of the header of the patch, containing the start bit, changed in the time of starting preparation of the patch for recording, the bit of correct writing of the size and format, changed after the data size or format are correctly recorded, the bit of correct writing, changed after the whole patch is correctly saved in the non-volatile memory, the invalidation bit, changed after the patch is invalidated and the field, defining the format of the patch and the field, defining the quantity of data in the patch, not considering the size of the header, saving of the patch starts from a change of the value of the start bit, after which the size and the type of the patch is recorded and if an error occurs the procedure ends, while if the writing of the fields is correct, the value of the bit in the field of correct writing of the size and format is changed and separate data groups are further recorded, and if an error occurs the procedure ends, while, if the so-far-writing is correct the value of the bit of correct writing is changed and at this moment the procedure ends and the patch is correctly recorded in the non-volatile memory.